

Git ve GitHub'a Giriş

Efe ÇİFTÇİ, Ph.D.

Çankaya Üniversitesi Bilgisayar Mühendisliği Bölümü

Ekim 2023



efeciftci.com



academic/~efeciftci



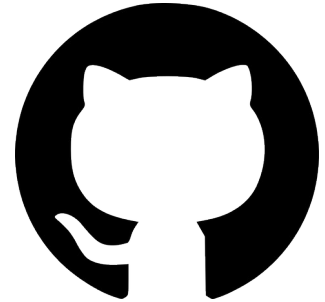
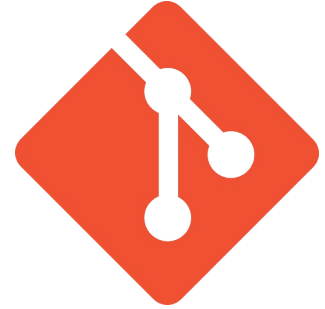
in/efeciftci

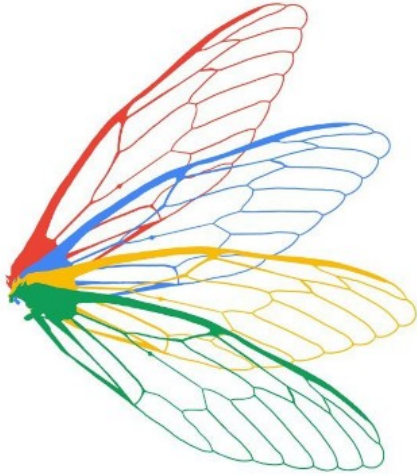


efeciftci



efeciftci





 Google Developer Student Clubs
Çankaya University



Çankaya University
Computer Engineering Society

Dur Şu Ödevin Bir Yedeğini Alayım...

Name	^	Size
<code>C++</code> odev.cpp		318 B
<code>C++</code> odevson.cpp		627 B
<code>C++</code> odevsonson.cpp		642 B
<code>C++</code> odevv.cpp		453 B
<code>C++</code> odevv2.cpp		491 B
<code>C++</code> odevvkopya.cpp		559 B

Sürüm Kontrol Sistemleri

- Sürüm kontrol sistemleri, yazılım geliştirmenin temel bir parçasıdır ve kod ile dökümanlar gibi kaynaklar üzerindeki değişiklikleri yöneten ve kontrol altında tutan yazılımlardır.
- Geliştiriciler, üzerinde çalıştıkları projelerdeki dosyaların farklı anlardaki durumlarını sürüm kontrol sistemleri üzerinde kayıt altına alabilirler.
- Bu kayıtlara “revizyon” denir.
- Bu sistem sayesinde herhangi bir anda eski revizyonlara geri dönüp bakılabilir, proje ve projeyi oluşturan dosyaların farklı revizyonları arasında karşılaştırma yapılabilir, veya proje eski bir revizyona geri döndürülebilir.

Sürüm Kontrol Sistemleri

- Aynı proje üzerinde birden fazla kişi çalışabilir.
- Sürüm kontrol sistemleri, yazılım geliştirme projelerini işbirliği yapılabilir ve yönetilebilir kılar.
- Geliştiricilerin eş zamanlı olarak aynı projede çalışabilmesini, projenin geçmiş durumlarına erişebilmesini ve değişiklikleri takip edebilmesini sağlar. Bu da yazılım geliştirme süreçlerini daha etkili ve düzenli hale getirir.
- Revision Control System (1982), Concurrent Versions System (1986), Visual SourceSafe (1994), Subversion (2000), Bitkeeper (2000), Git (2005), Mercurial (2005) ...

Örnek: Hesap Makinesi

improved readability efeciftci committed 5 hours ago
added divbyzero check ayhanarici committed 5 hours ago
added division efeciftci committed 5 hours ago
simplified variables efeciftci committed 6 hours ago
added multiplication ayhanarici committed 6 hours ago
added difference variable efeciftci committed 6 hours ago
added difference calculation efeciftci committed 6 hours ago
initial version efeciftci committed 6 hours ago

```
10 main.c
...
1 + #include <stdio.h>
2 +
3 + int main() {
4 +     int a, b, sum;
5 +     printf("Please enter a and b: ");
6 +     scanf("%d %d", &a, &b);
7 +     sum = a + b;
8 +     printf("Sum of a and b is %d\n", sum);
9 +     return 0;
10 + }
```

Örnek: Hesap Makinesi

improved readability efeciftci committed 5 hours ago
added divbyzero check ayhanarici committed 5 hours ago
added division efeciftci committed 5 hours ago
simplified variables efeciftci committed 6 hours ago
added multiplication ayhanarici committed 6 hours ago
added difference variable efeciftci committed 6 hours ago
added difference calculation efeciftci committed 6 hours ago
initial version efeciftci committed 6 hours ago

```
main.c  
-6,5 +6,7  
int main() {  
    scanf("%d %d", &a, &b);  
    sum = a + b;  
    printf("Sum of a and b is %d\n", sum);  
    sum = a - b;  
    printf("Difference between a and b is %d\n", sum);  
    return 0;  
}
```

Örnek: Hesap Makinesi

improved readability efeciftci committed 5 hours ago
added divbyzero check ayhanarici committed 5 hours ago
added division efeciftci committed 5 hours ago
simplified variables efeciftci committed 6 hours ago
added multiplication ayhanarici committed 6 hours ago
added difference variable efeciftci committed 6 hours ago
added difference calculation efeciftci committed 6 hours ago
initial version efeciftci committed 6 hours ago

```
6 main.c
... -1,12 +1,12 @@
1 1 #include <stdio.h>
2 2
3 3 int main() {
4 - int a, b, sum;
4 + int a, b, sum, difference;
5 5 printf("Please enter a and b: ");
6 6 scanf("%d %d", &a, &b);
7 7 sum = a + b;
8 8 printf("Sum of a and b is %d\n", sum);
9 - sum = a - b;
10 - printf("Difference between a and b is %d\n", sum);
9 + difference = a - b;
10 + printf("Difference between a and b is %d\n", difference);
11 11 return 0;
12 12 }
```


Örnek: Hesap Makinesi

improved readability efeciftci committed 5 hours ago
added divbyzero check ayhanarici committed 5 hours ago
added division efeciftci committed 5 hours ago
simplified variables efeciftci committed 6 hours ago
added multiplication ayhanarici committed 6 hours ago
added difference variable efeciftci committed 6 hours ago
added difference calculation efeciftci committed 6 hours ago
initial version efeciftci committed 6 hours ago

```
4 main.c
... -1,12 +1,14 @@
1 1 #include <stdio.h>
2 2
3 3 int main() {
4 - int a, b, sum, difference;
4 + int a, b, sum, difference, multiplication;
5 5 printf("Please enter a and b: ");
6 6 scanf("%d %d", &a, &b);
7 7 sum = a + b;
8 8 printf("Sum of a and b is %d\n", sum);
9 9 difference = a - b;
10 10 printf("Difference between a and b is %d\n", difference);
11 + multiplication=a*b;
12 + printf("Multiplication of a and b is %d\n", multiplication);
11 13 return 0;
12 14 }
```

Örnek: Hesap Makinesi

improved readability efeciftci committed 5 hours ago
added divbyzero check ayhanarici committed 5 hours ago
added division efeciftci committed 5 hours ago
simplified variables efeciftci committed 6 hours ago
added multiplication ayhanarici committed 6 hours ago
added difference variable efeciftci committed 6 hours ago
added difference calculation efeciftci committed 6 hours ago
initial version efeciftci committed 6 hours ago

```
main.c
... -1,14 +1,14 @@
1 1 #include <stdio.h>
2 2
3 3 int main() {
4 - int a, b, sum, difference, multiplication;
4 + int a, b, result;
5 5 printf("Please enter a and b: ");
6 6 scanf("%d %d", &a, &b);
7 - sum = a + b;
8 - printf("Sum of a and b is %d\n", sum);
9 - difference = a - b;
10 - printf("Difference between a and b is %d\n", difference);
11 - multiplication=a*b;
12 - printf("Multiplication of a and b is %d\n", multiplication);
7 + result = a + b;
8 + printf("Sum of a and b is %d\n", result);
9 + result = a - b;
10 + printf("Difference between a and b is %d\n", result);
11 + result=a*b;
12 + printf("Multiplication of a and b is %d\n", result);
13 13 return 0;
14 14 }
```

Örnek: Hesap Makinesi

improved readability efeciftci committed 5 hours ago
added divbyzero check ayhanarici committed 5 hours ago
added division efeciftci committed 5 hours ago
simplified variables efeciftci committed 6 hours ago
added multiplication ayhanarici committed 6 hours ago
added difference variable efeciftci committed 6 hours ago
added difference calculation efeciftci committed 6 hours ago
initial version efeciftci committed 6 hours ago

```
5 main.c
↑... @@ -2,13 +2,16 @@
2 2
3 3 int main() {
4 4     int a, b, result;
5 5 +     float divresult;
5 6     printf("Please enter a and b: ");
6 7     scanf("%d %d", &a, &b);
7 8     result = a + b;
8 9     printf("Sum of a and b is %d\n", result);
9 10    result = a - b;
10 11    printf("Difference between a and b is %d\n", result);
11 11 -    result=a*b;
12 12 +    result = a * b;
12 13    printf("Multiplication of a and b is %d\n", result);
14 14 +    divresult = (float) a / b;
15 15 +    printf("Division of a by b is %f\n", divresult);
13 16    return 0;
14 17 }
```

Örnek: Hesap Makinesi

improved readability efeciftci committed 5 hours ago
added divbyzero check ayhanarici committed 5 hours ago
added division efeciftci committed 5 hours ago
simplified variables efeciftci committed 6 hours ago
added multiplication ayhanarici committed 6 hours ago
added difference variable efeciftci committed 6 hours ago
added difference calculation efeciftci committed 6 hours ago
initial version efeciftci committed 6 hours ago

```
8 main.c
...
11 11      printf("Difference between a and b is %d\n", result);
12 12      result = a * b;
13 13      printf("Multiplication of a and b is %d\n", result);
14 -      divresult = (float) a / b;
15 -      printf("Division of a by b is %f\n", divresult);
14 +      if(b==0)
15 +          printf("You cannot divide by zero!");
16 +      else {
17 +          divresult = (float) a / b;
18 +          printf("Division of a by b is %f\n", divresult);
19 +      }
16 20      return 0;
17 21  }
```

Git Tarihçesi



- 2005 yılı öncesinde Linux işletim sistemi çekirdeği, kapalı kaynak kodlu olan BitKeeper sistemi üzerinde geliştirilmekteydi.
- BitKeeper'ın sahibi olan firmanın, Linux geliştiricilerine tanıdığı ücretsiz kullanım iznini iptal etmesi üzerine Linus Torvalds ve diğer çekirdek geliştiricileri kendi sürüm kontrol sistemlerini geliştirmeye başladılar.
- Bu yeni sürüm kontrol sistemi Git adını alır.

Git Kullanımı

- Projelerinizi Git ile yönetmeye başlamanın birden fazla yolu var:
 - Bilgisayardaki proje dizininde yerel bir Git deposu oluşturabilirsiniz,
 - Şirket gibi ortamlarda tüm çalışanların ortak erişebileceği bir Git sunucusu kurabilirsiniz,
 - GitHub gibi dünyaya açık hizmetlerden faydalanabilirsiniz.

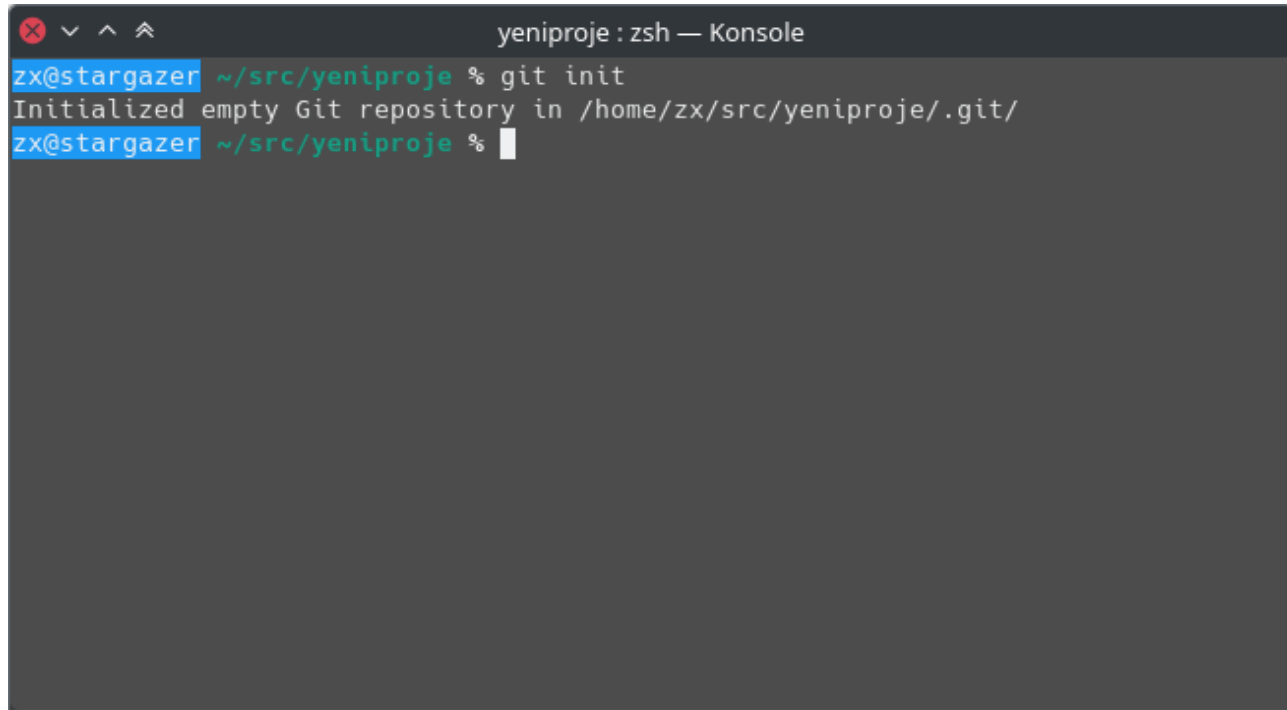
<https://git-scm.com/downloads>



- Git'i hem komut satırından, hem görsel uygulamalardan kullanmak mümkündür.

Yeni Git Deposu Oluşturma

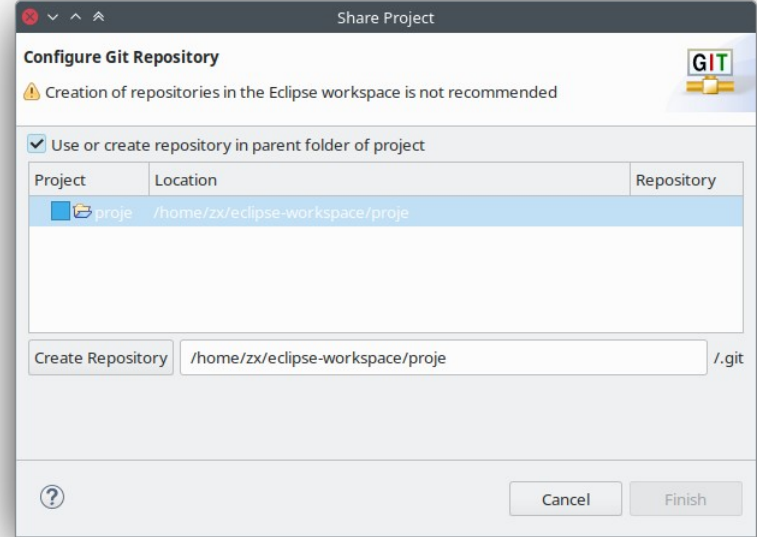
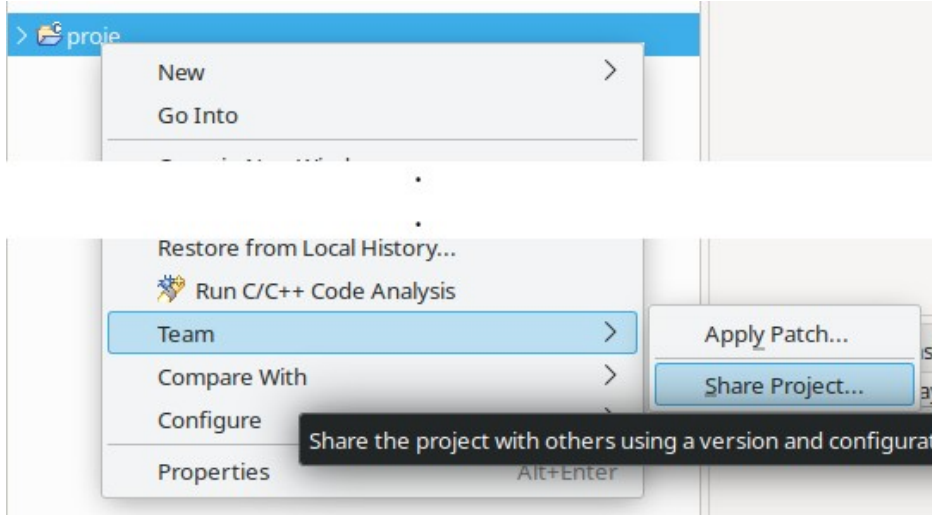
- `git init` komutu ile proje dizininizde yeni bir Git deposu oluşturabilirsiniz.



```
yeniproje : zsh — Konsole
zx@stargazer ~/src/yeniproje % git init
Initialized empty Git repository in /home/zx/src/yeniproje/.git/
zx@stargazer ~/src/yeniproje %
```

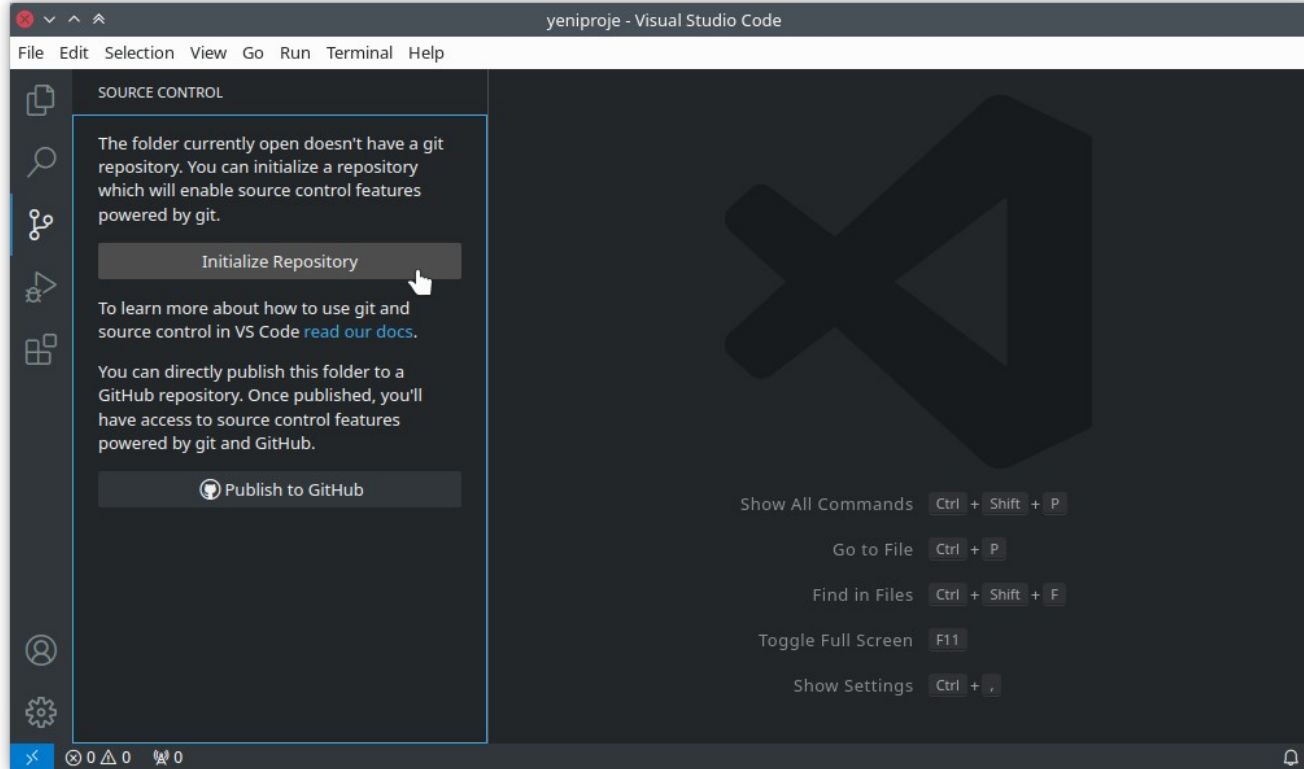
Yeni Git Deposu Oluřturma

- Grafik arayüz ile de mümkün; örneğın Eclipse:



Yeni Git Deposu Oluřturma

- Grafik arayüz ile de mümkün; örneğın Visual Studio Code:



Dosya Ekleme, Taşıma, Silme

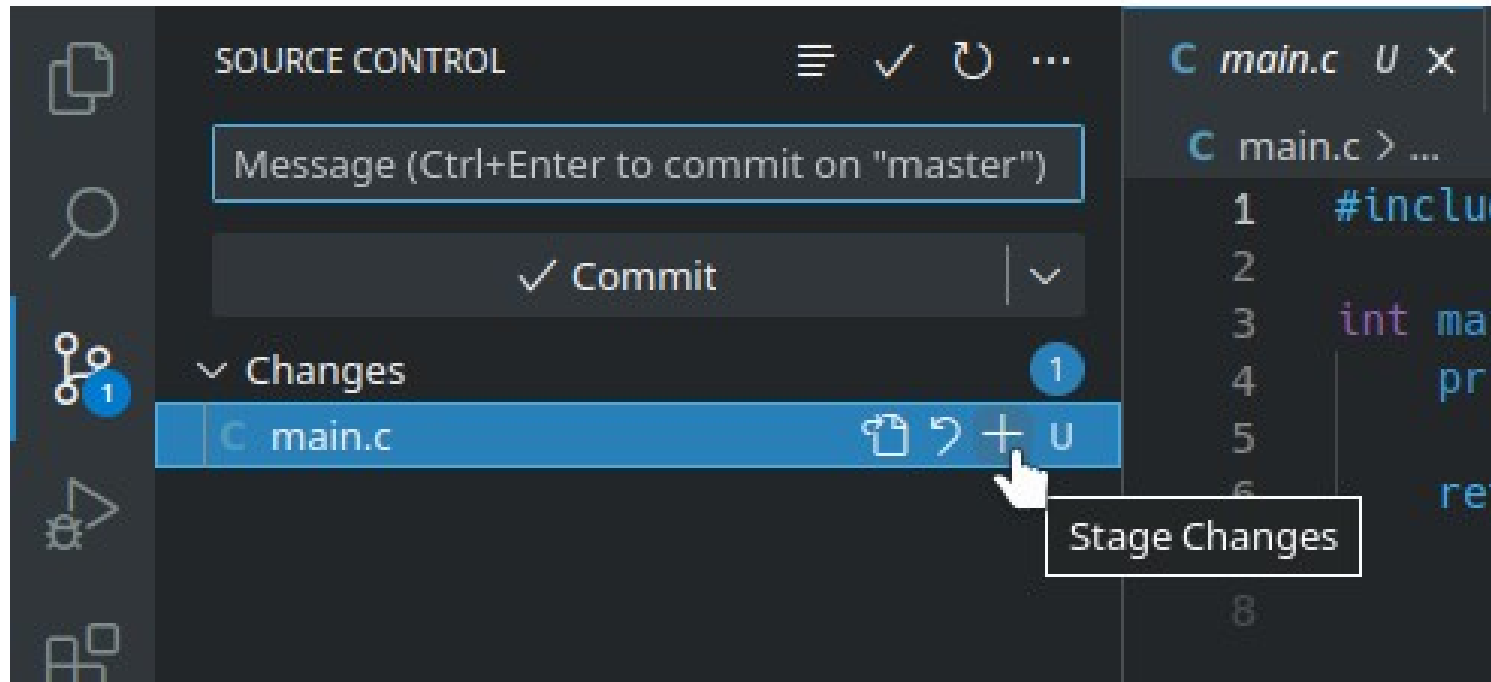
- Proje ağacındaki bir sonraki revizyona;
 - Yeni dosya eklemek veya mevcut dosyalarda yapılan değişiklikleri tanıtmak için `git add` komutu,
 - Bir dosyayı başka bir dizine taşımak veya adını değiştirmek için `git mv` komutu,
 - Bir dosyayı proje ağacından çıkartmak için `git rm` komutu

kullanılır.

Dosya Ekleme, Taşıma, Silme

```
x v ^ ^
proje : zsh — Konsole
zx@voyager ~/proje % git ls-tree --name-only HEAD
fact.c
main
main.c
zx@voyager ~/proje % ls
fact.c main main.c todo.txt
zx@voyager ~/proje % git rm main
rm 'main'
zx@voyager ~/proje % git add main.c
zx@voyager ~/proje % git mv fact.c factorial.c
zx@voyager ~/proje % git add todo.txt
zx@voyager ~/proje % git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    renamed:    fact.c -> factorial.c
    deleted:    main
    modified:   main.c
    new file:   todo.txt
zx@voyager ~/proje % _
```

Dosya Ekleme, Taşıma, Silme



Revizyon Oluşturma

- Yeni revizyon için tüm ekleme/çıkartma/taşıma işlemleri bittiğinde `git commit` komutu ile revizyon oluşturulur.
- Her revizyonun bir açıklaması olması gereklidir.
 - `git commit` bu açıklamayı harici bir metin editöründen okur,
 - `git commit -m MESAJ` kısa açıklamaların hızlıca komut satırından girilmesini sağlar.

Revizyon Oluşturma

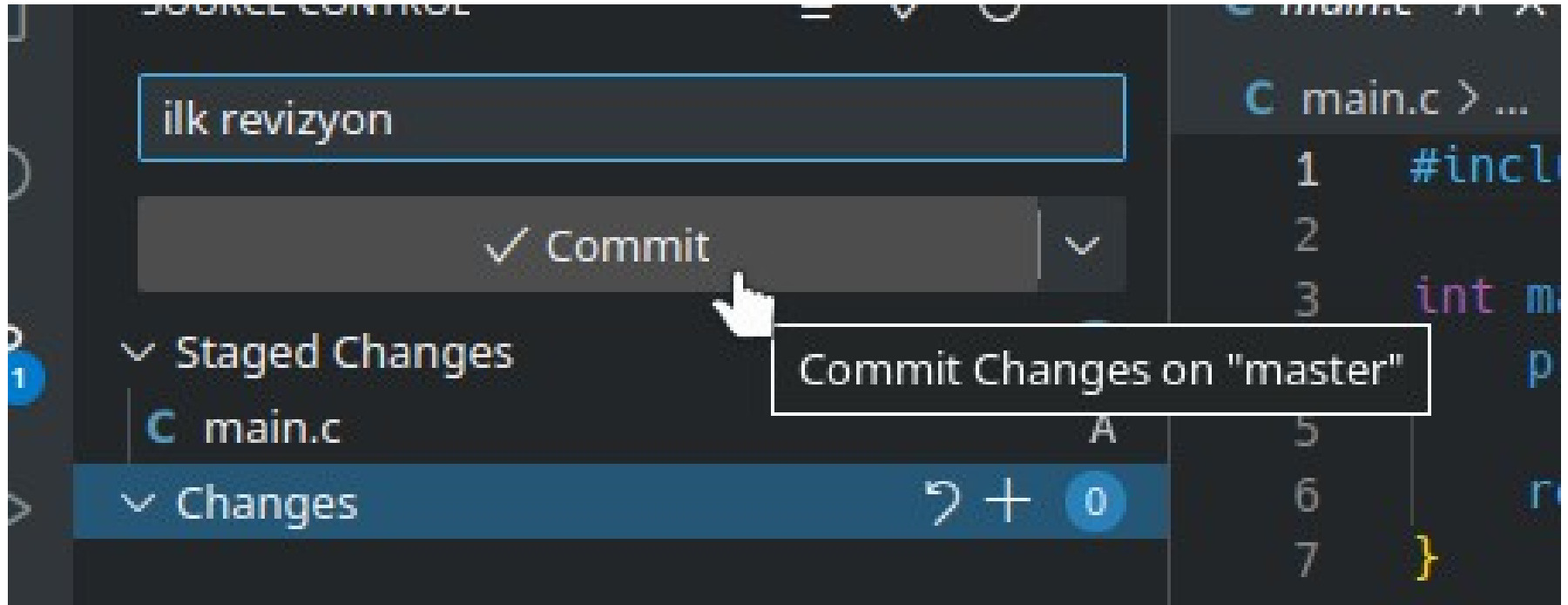
```
yeniproje : zsh — Konsole
zx@stargazer ~/src/yeniproje % git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   main.c

zx@stargazer ~/src/yeniproje % git commit -m 'ilk revizyon'
[master (root-commit) b90fc67] ilk revizyon
 1 file changed, 7 insertions(+)
 create mode 100644 main.c
zx@stargazer ~/src/yeniproje %
```

Revizyon Oluşturma



Revizyonlar Arası Karşılaştırma

- `git log` komutu ile projeye ait tüm revizyonlar listelenebilir:

```
gitpresentation2023 : zsh — Konsole
zx@enterprise ~/gitpresentation2023
% git log
commit 824e9bc11a23b99cb8c939c96b424b48e3dd15d5 (HEAD -> master, origin/master, origin/HEAD)
Author: Efe Çiftci <[REDACTED]>
Date:   Wed Oct 11 11:37:39 2023 +0300

    improved readability

commit 07e444b121ade1f973206957b03ca7a183054d41 (origin/division-functionality)
Author: Ayhan ARICI <[REDACTED]>
Date:   Wed Oct 11 11:13:19 2023 +0300

    added divbyzero check

commit 8f92e7e07b0f03950ccae741ac1270260e172b6f
Author: Efe Çiftci <[REDACTED]>
Date:   Wed Oct 11 11:01:58 2023 +0300

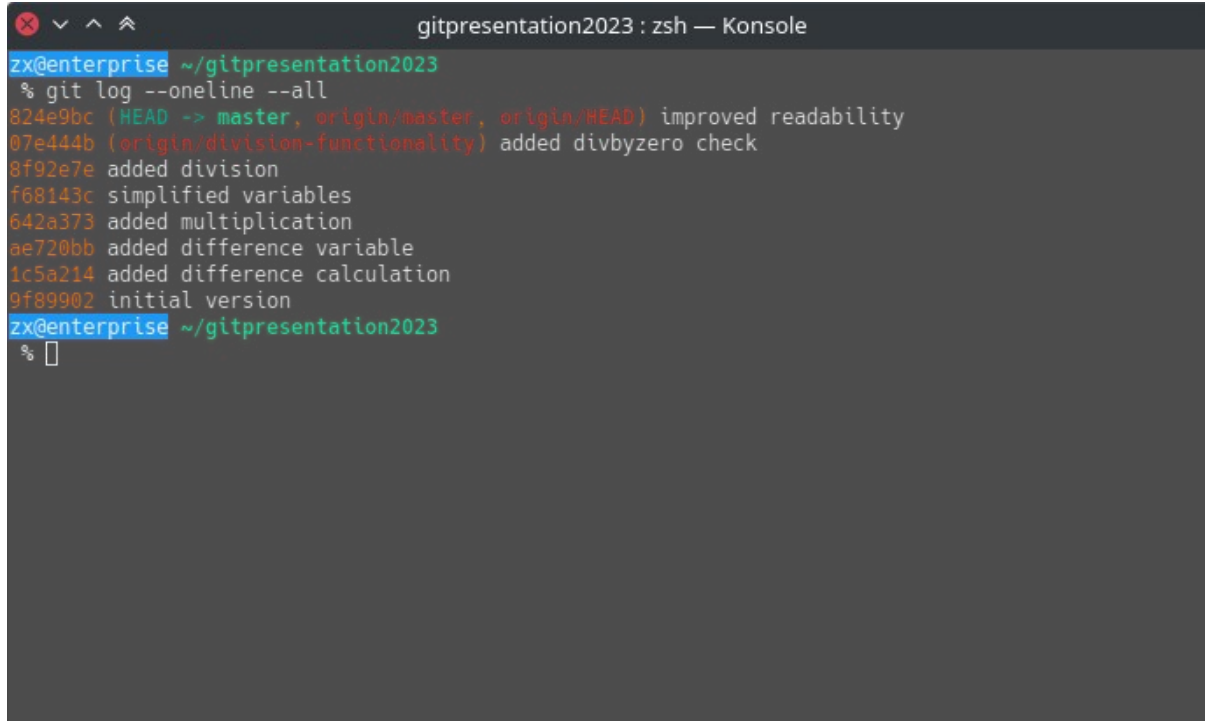
    added division

commit f68143c34294c9cc098a25267298b1b0130a16ef
Author: Efe Çiftci <[REDACTED]>
Date:   Wed Oct 11 10:35:07 2023 +0300

    simplified variables
```


Revizyonlar Arası Karşılaştırma

- `git log --oneline` komutu ile projeye ait tüm revizyonlar özet olarak listelenebilir:

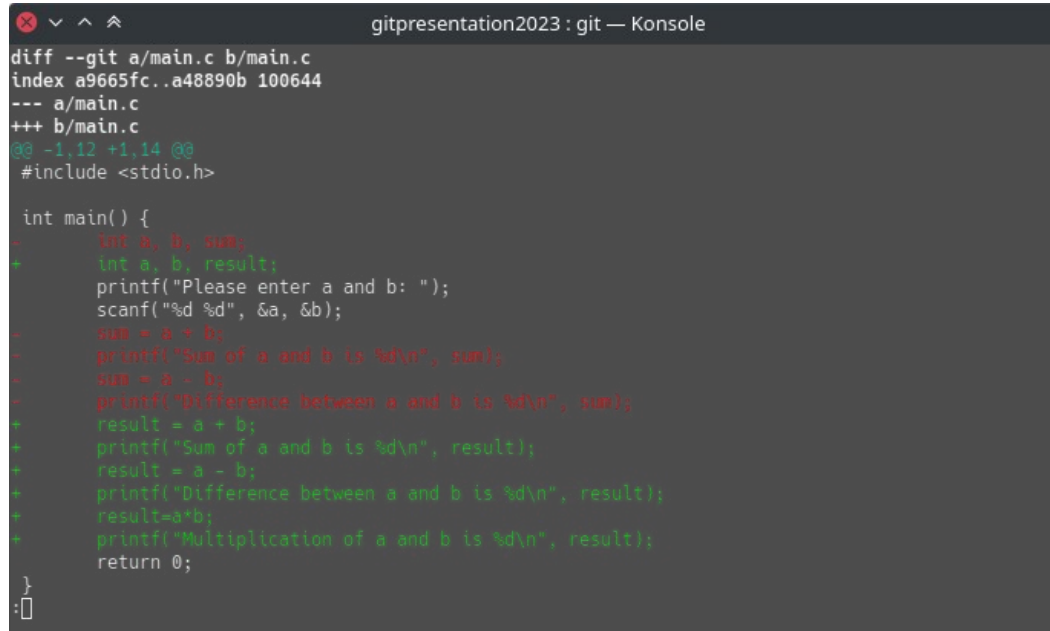
A terminal window titled "gitpresentation2023 : zsh — Konsole" showing the execution of the command "git log --oneline --all". The output lists seven commits with their hashes and descriptions. The terminal prompt is "zx@enterprise ~/gitpresentation2023".

```
zx@enterprise ~/gitpresentation2023
% git log --oneline --all
824e9bc (HEAD -> master, origin/master, origin/HEAD) improved readability
07e444b (origin/division-functionality) added divbyzero check
8f92e7e added division
f68143c simplified variables
642a373 added multiplication
ee720bb added difference variable
1c5a214 added difference calculation
9f89902 initial version
zx@enterprise ~/gitpresentation2023
% □
```

Revizyonlar Arası Karşılaştırma

- `git diff REV1 REV2` komutu ile revizyon numarası verilen iki revizyon arasındaki farklılıklar listelenebilir.

```
git diff 1c5a214 f68143c
```



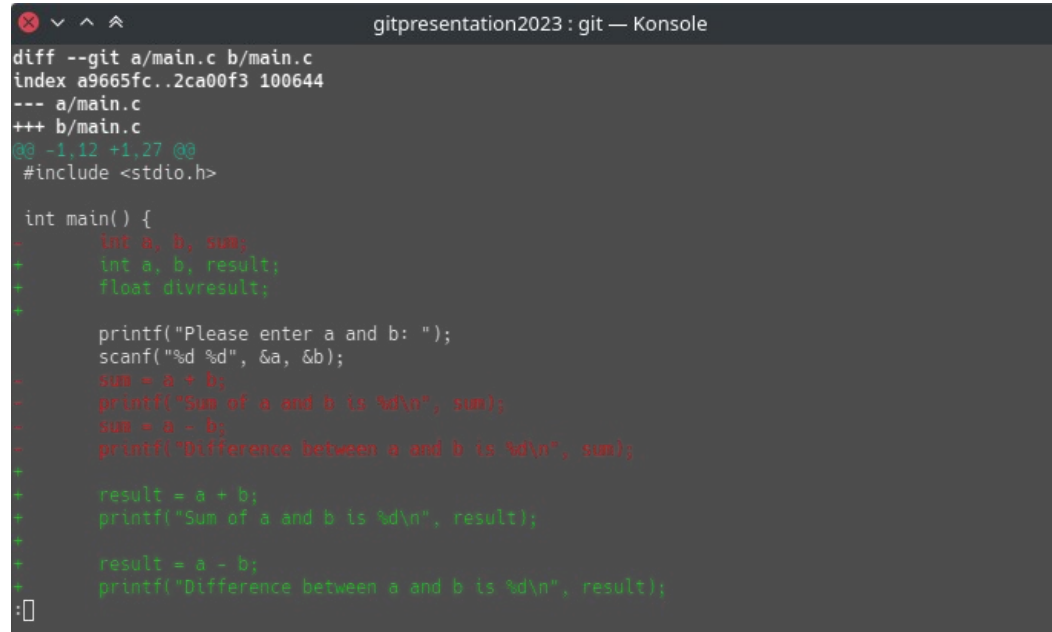
```
diff --git a/main.c b/main.c
index a9665fc..a48890b 100644
--- a/main.c
+++ b/main.c
@@ -1,12 +1,14 @@
#include <stdio.h>

int main() {
-   int a, b, sum;
+   int a, b, result;
   printf("Please enter a and b: ");
   scanf("%d %d", &a, &b);
-   sum = a + b;
-   printf("Sum of a and b is %d\n", sum);
-   sum = a - b;
-   printf("Difference between a and b is %d\n", sum);
+   result = a + b;
+   printf("Sum of a and b is %d\n", result);
+   result = a - b;
+   printf("Difference between a and b is %d\n", result);
+   result=a*b;
+   printf("Multiplication of a and b is %d\n", result);
   return 0;
}
```

Revizyonlar Arası Karşılaştırma

- Birden fazla dosyanın değiştiği revizyonlarda sadece belli dosyalardaki farklılıkları görüntülemek için `git diff REV1 REV2 DOSYA` komutu kullanılabilir.

```
git diff 1c5a214 824e9bc main.c
```



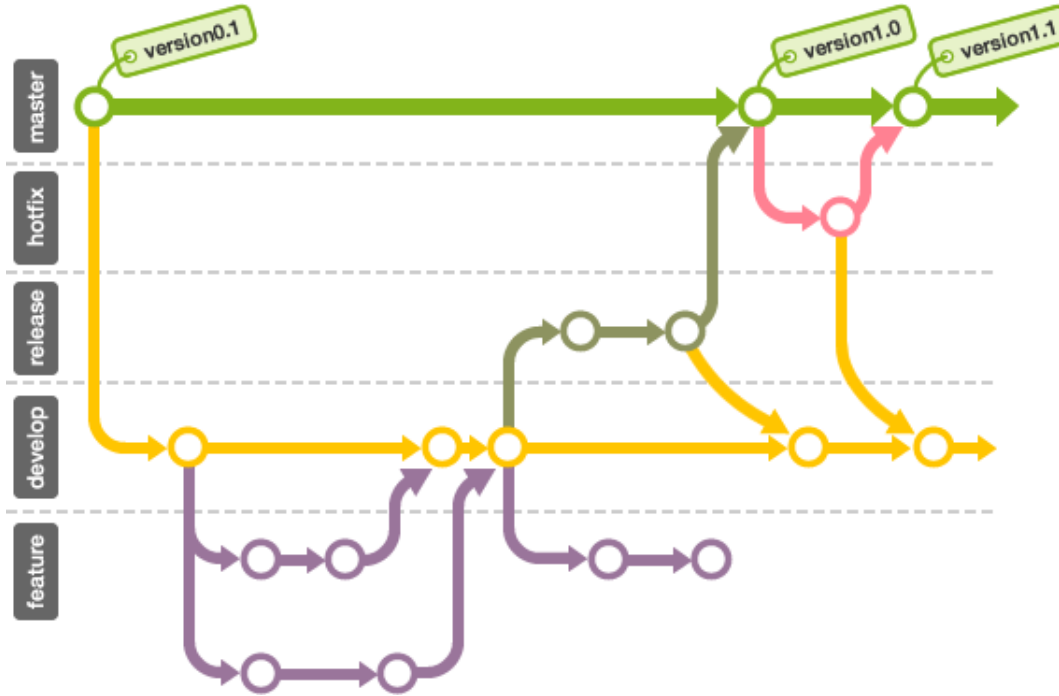
```
diff --git a/main.c b/main.c
index a9665fc..2ca0f3 100644
--- a/main.c
+++ b/main.c
@@ -1,12 +1,27 @@
#include <stdio.h>

int main() {
-   int a, b, sum;
+   int a, b, result;
+   float divresult;
+
   printf("Please enter a and b: ");
   scanf("%d %d", &a, &b);
-   sum = a * b;
-   printf("Sum of a and b is %d\n", sum);
-   sum = a - b;
-   printf("Difference between a and b is %d\n", sum);
+
+   result = a + b;
+   printf("Sum of a and b is %d\n", result);
+
+   result = a - b;
+   printf("Difference between a and b is %d\n", result);
}
```

Branch Kavramı

- Git ile yönetilen projeler üzerinde farklı dallar (branch) oluşturulabilir.
- Bu dallar, projenin farklı özelliklerini veya hata düzeltmelerini geliştirmek için kullanılır.
- Her dal, projenin bağımsız bir kopyasını temsil eder ve geliştiriciler farklı görevler üzerinde çalışırken ana projeyi etkilemeden değişiklik yapabilirler.
- İş tamamlandığında veya bir dalın geliştirmesi sona erdiğinde bu dallar ana projeye birleştirilebilir.

Branch Kavramı



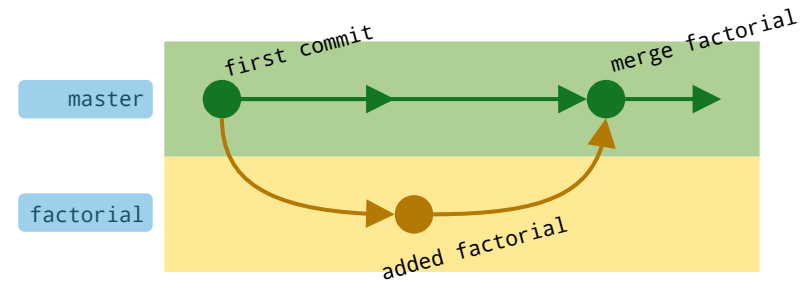
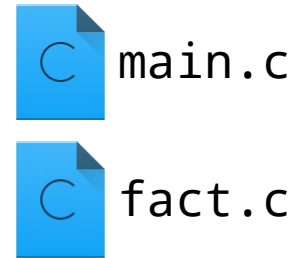
Kaynak: Simple Git tutorial for beginners

Branch Kavramı

- `git branch` ile mevcut dallar görüntülenir.
- `git branch İSİM` ile yeni bir dal oluşturulur.
- `git checkout İSİM` ile dallar arasında geçiş yapılır.
- `git checkout -b İSİM` ile yeni bir dal oluşturulur ve bu yeni dala geçiş yapılır.
- `git merge İSİM`, İSİM dalını mevcut dal ile birleştirir.
- `git merge --no-ff -m MESAJ İSİM`, İSİM dalını mevcut dal ile MESAJ açıklamalı bir revizyon oluşturarak birleştirir.

Branch Kavramı

```
$ git init
Initialized empty Git repository in /home/zx/proje/.git/
$ git add main.c
$ git commit -m 'first commit'
[master (root-commit) 9ec86ad] first commit
1 file changed, 4 insertions(+)
create mode 100644 main.c
$ git checkout -b factorial
Switched to a new branch 'factorial'
$ git add main.c fact.c
$ git commit -m 'added factorial'
[factorial 1536825] added factorial
2 files changed, 7 insertions(+)
create mode 100644 fact.c
$ git checkout master
Switched to branch 'master'
$ git merge --no-ff -m 'merge factorial' factorial
Merge made by the 'ort' strategy.
fact.c | 6 ++++++
main.c | 1 +
2 files changed, 7 insertions(+)
create mode 100644 fact.c
$ git log --oneline --graph
* 36fdcd4 (HEAD -> master) merge factorial
|
| * 1536825 (factorial) added factorial
|/
* 9ec86ad first commit
```



GitHub

- Github, küresel bir Git proje depolama hizmetidir.

<https://github.com/>

- Projelerinizi çevrimiçi olarak depolamanıza ve yönetmenize olanak tanır.
- Projelerinizi diğer geliştiricilerle paylaşabilir ve işbirliği yapabilirsiniz.
- Ücretsiz bir GitHub hesabı oluşturarak projelerinizi oluşturabilir ve bunların görünürlüğünü dilediğiniz gibi ayarlayabilirsiniz.
- Diğer geliştiricilere ait projeleri inceleyebilir, bu projelerin kopyalarını oluşturabilir ve hata düzeltmeleri veya yeni özellikler ekleyebilirsiniz.

GitHub Üzerinde Proje Yönetimi

The screenshot displays the GitHub Dashboard for a user named 'efciftci'. The interface is divided into several sections:

- Header:** Includes the GitHub logo, the word 'Dashboard', a search bar with the placeholder 'Type to search', and navigation icons for home, repositories, activity, and profile.
- Left Sidebar:**
 - efciftci** profile name with a dropdown arrow.
 - Top Repositories:** A list of repositories with a search input 'Find a repository...' and a green 'New' button highlighted by a red arrow.
 - Recent activity:** A dashed box containing the text: 'When you take actions across GitHub, we'll provide links to that activity here.'
- Main Content Area:**
 - Home:** A central message that says 'That's all for now' and 'You can adjust your [feed filter](#) to see more content, or visit [Explore](#).'
 - Latest changes:** A list of recent updates, including 'New branches and commits pages - feature preview', 'Copilot October 23rd Update', and 'Authress is now a GitHub secret scanning partner'.
 - Explore repositories:** A list of featured repositories, including 'microsoft / AirSim' (15.2k stars, C++) and 'jenkinsci / docker' (6.2k stars, Dockerfile).

GitHub Üzerinde Proje Yönetimi

New repository


Q Type to search

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Required fields are marked with an asterisk ().*


Owner * **Repository name ***

 efeciftci /

Great repository names are short and memorable. Need inspiration? How about [crispy-goggles](#) ?

Description (optional)

 **Public**
Anyone on the internet can see this repository. You choose who can commit.

 **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

GitHub Üzerinde Proje Yönetimi

Get started with GitHub Copilot

Invite collaborators

Quick setup — if you've done this kind of thing before

or HTTPS SSH

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M master
git remote add origin git@github.com:efeciftci/demo.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin git@github.com:efeciftci/demo.git
git branch -M master
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

GitHub Üzerinde Proje Yönetimi

- Var olan bir proje `git clone UZAK_REPO_ADRESİ` komutu ile projenin tüm geçmişi ile birlikte bilgisayara indirilebilir.

```
gitpresentation2023: zsh — Konsole
zx@enterprise ~ % git clone git@github.com:efeciftci/gitpresentation2023.git
Cloning into 'gitpresentation2023'...
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 24 (delta 7), reused 16 (delta 3), pack-reused 0
Receiving objects: 100% (24/24), done.
Resolving deltas: 100% (7/7), done.
zx@enterprise ~ % cd gitpresentation2023
zx@enterprise ~/gitpresentation2023
% ls -l
total 4
-rw-rw-r-- 1 zx zx 493 Oct 24 20:23 main.c
zx@enterprise ~/gitpresentation2023
% git log --oneline
824e9bc (HEAD -> master, origin/master, origin/HEAD) improved readability
07e444b (origin/division-functionality) added divbyzero check
8f92e7e added division
f68143c simplified variables
642a373 added multiplication
ae720bb added difference variable
1c5a214 added difference calculation
9f89902 initial version
zx@enterprise ~/gitpresentation2023
% □
```

GitHub Üzerinde Proje Yönetimi

- Kendi bilgisayarınızda oluşturduğunuz revizyonlar karşı sunucuya **git push** komutu ile iletilir.

```
demo2023: zsh — Konsole
zx@voyager ~/Desktop/demo2023 % git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   main.c

no changes added to commit (use "git add" and/or "git commit -a")
zx@voyager ~/Desktop/demo2023 % git add main.c
zx@voyager ~/Desktop/demo2023 % git commit -m 'fixed typo'
[master 214cd2c] fixed typo
 1 file changed, 1 insertion(+)
zx@voyager ~/Desktop/demo2023 % git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 271 bytes | 271.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:efeciftci/demo2023.git
   c62adea..214cd2c master -> master
zx@voyager ~/Desktop/demo2023 % _
```

GitHub Üzerinde Proje Yönetimi

- Bir projenin bilgisayarda tutulan kopyasını, uzak sunucudaki güncel revizyona çekmek için `git pull` komutu kullanılır.

```
gitpresentation2023 : zsh — Konsole
zx@voyager ~/Desktop/gitpresentation2023
% git log --oneline
f68143c (HEAD -> master) simplified variables
642a373 added multiplication
ae720bb added difference variable
1c5a214 added difference calculation
9f89902 initial version
zx@voyager ~/Desktop/gitpresentation2023
% git pull
Updating f68143c..824e9bc
Fast-forward
 main.c | 15 ++++++++
 1 file changed, 14 insertions(+), 1 deletion(-)
zx@voyager ~/Desktop/gitpresentation2023
% git log --oneline
824e9bc (HEAD -> master, origin/master, origin/HEAD) improved readability
07e444b (origin/division-functionality) added divbyzero check
8f92e7e added division
f68143c simplified variables
642a373 added multiplication
ae720bb added difference variable
1c5a214 added difference calculation
9f89902 initial version
zx@voyager ~/Desktop/gitpresentation2023
% _
```

GitHub Üzerinde Proje Yönetimi

- Projelerin başarılı bir şekilde yönetilmesi ve kullanıcıların projeyi anlamaları için dökümantasyon önemlidir.
- GitHub, Markdown dilini kullanarak proje dökümantasyonunuzu kolayca oluşturmanızı sağlar.
 - README.md
 - Changelog.md
 - License.md
 - ...

GitHub Üzerinde Proje Yönetimi

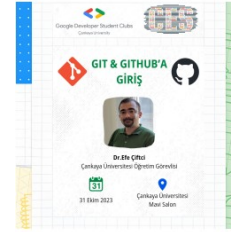
```
1 # Git ve GitHub'a Giriş
2 Çankaya Üniversitesi Bilgisayar Mühendisliği Bölümü üyesi Öğr. Gör. Dr. Efe
3 Çiftci tarafından verilecek olan bu seminerde katılımcılarla aşağıdaki
4 konular hakkında bilgiler paylaşılacaktır:
5 - Sürüm kontrol sistemleri
6 - Git tarihçesi
7 - En sık kullanılan Git özellikleri
8   - Depo oluşturma
9   - Revizyon oluşturma
10  - Branch oluşturma ve birleştirme
11  - ve daha fazlası!
12
13 ![poster](assets/poster.png)
14
15 **Tarih:** 31 Ekim 2023
16 **Saat:** 13:30
17 **Yer:** Merkez Kampüs Mavi Salon
18
19 ## Slayt Notları
20 Sunum sırasında kullanılacak olan slaytlara [konuşmacı kişisel web
21 sayfasından](https://efeciftci.com/) erişebilirsiniz.
```



Git ve GitHub'a Giriş

Çankaya Üniversitesi Bilgisayar Mühendisliği Bölümü üyesi Öğr. Gör. Dr. Efe Çiftci tarafından verilecek olan bu seminerde katılımcılarla aşağıdaki konular hakkında bilgiler paylaşılacaktır:

- Sürüm kontrol sistemleri
- Git tarihçesi
- En sık kullanılan Git özellikleri
 - Depo oluşturma
 - Revizyon oluşturma
 - Branch oluşturma ve birleştirme
 - ve daha fazlası!



Tarih: 31 Ekim 2023

Saat: 13:30

Yer: Merkez Kampüs Mavi Salon

Slayt Notları

Sunum sırasında kullanılacak olan slaytlara [konuşmacı kişisel web sayfasından](https://efeciftci.com/) erişebilirsiniz.

Sorularınız?

Dinlediđiniz İin Teřekkürler!

*Bu sunum özgür iřletim sistemi [KDE neon](#)
üzerinde özgür ofis yazılımı [LibreOffice](#)
ile üretilmiřtir.*

"Free as in free speech, not free beer"

